

(54) MANAGEMENT SYSTEM FOR OF NUMBER OF TIMES OF WRITING
PROGRAMMABLE READ ONLY MEMORY

(11) 62-283487 (A) (43) 9.12.1987 (19) JP

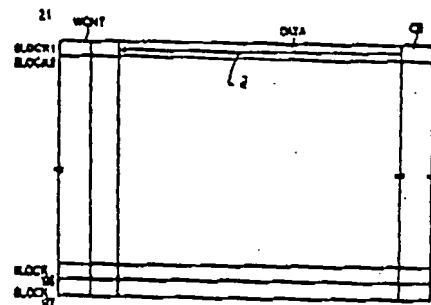
(21) Appl. No. 61-124732 (22) 31.5.1986

(71) CANON INC (72) SHINICHI NAKADA

(51) Int. Cl. G11C17/00

PURPOSE: To average the rewriting frequency of a memory block and to prolong the life of an EEPROM by suppressing the writing to the memory block reaching the number of times of setting.

CONSTITUTION: The memory area of the EEPROM in which the erasing, the rewriting or the like are carried out by an input means, a CPU or the like is divided into 127 such as blocks BLOCK1~BLOCK127 and in the respective blocks, an area WCNT for storing the updating and rewriting number of times as well as a data memory area DATA is provided. When the contents of the area WCNT in which the counted value of the updating counter of the directory area of a block pointer is written are referred to and reach the set number, the rewriting of the block is suppressed through the CPU. The rewriting frequency of the respective blocks is averaged and the life of the EEPROM is prolonged.



a: storage data

⑨ 日本国特許庁(JP)

⑩ 特許出願公開

⑫ 公開特許公報(A)

昭62-283497

⑬ Int. Cl.⁴

識別記号

庁内整理番号

⑭ 公開 昭和62年(1987)12月9日

G 11 C 17/00

3 0 7

6549-5B

審査請求 未請求 発明の数 1 (全8頁)

⑮ 発明の名称 プログラマブルリードオンリメモリの書き込み回数管理方式

⑯ 特 願 昭61-124732

⑰ 出 願 昭61(1986)5月31日

⑱ 発 明 者 仲 田 真 一 東京都大田区下丸子3丁目30番2号 キャノン株式会社内
⑲ 出 願 人 キャノン株式会社 東京都大田区下丸子3丁目30番2号
⑳ 代 理 人 弁理士 小林 将高

明 細 書

1. 発明の名称

プログラマブルリードオンリメモリの書き込み
回数管理方式

2. 特許請求の範囲

記憶領域に書き込まれた情報を電気的に消去可能なプログラマブルリードオンリメモリにおいて、前記記憶領域を複数のブロックに分割し、各ブロック毎に書き込み回数を記憶し、あらかじめ設定される書き込み回数を越えたブロックへの書き込みを抑制させることを特徴とするプログラマブルリードオンリメモリの書き込み回数管理方式。

3. 発明の詳細な説明

(産業上の利用分野)

この発明は、電気的消去可能なプログラマブルリードオンリメモリの書き込み回数の管理方式に関するものである。

(従来の技術)

従来のEEPROM(Electrical Erasable

and Programmable ROM)は、容量も少なく、また書き込むために必要な外部回路が多かった。さらに、チップ内のすべてのデータを消去するモードしか有していなかった。最近では、容量も大きくなるとともに、外部回路も殆ど必要なくCPUのアドレスバス、データバスに接続できるようになり、またEEPROM内の1バイトのデータのみのも消去も可能となってきた。以上の改良により、使用目的によっては、従来のランダムアクセスメモリ(RAM)で構成していた機能の置換が可能となった。

例えば、従来の小型パソコン、日本語ワープロで作成したプログラムや文庫、外字等を保存しておくためにメモリカードと云うものがある。これは、必要なときにパソコン、日本語ワープロ等の本体に差し込んでプログラムや文庫を記憶させ、本体から引き抜いても、そのデータを記憶しているように、メモリカード内にはRAMと電池が搭載されていた。そこで、メモリカードをEEPROMで構成することにより、電池を無くすること

ができると考えられた。

(発明が解決しようとする問題点)

ところが、EEPROMでは従来のRAMのように自由に何度も書き換えられない制約があり、すなわち、あらかじめ設定される書き込み回数を越えて、メモリカードへの書き込みを行なうことにより、記憶しているはずのデータを消失させてしまう等の問題点があった。またEEPROMに書き込まれたデータのうち、頻りに書き換えられるデータと書き換え頻度の少ないデータとが存在し、書き換え頻度の高いデータの書き換え回数が所定値を越えると、EEPROMへの書き換えが可能にも関わらず書き換え不能となる問題点があった。

この発明は、上記の問題点を解消するためになされたもので、EEPROMに書き込まれるデータの消失を防止するとともに、EEPROMへの書き込み回数を平均化させるとともに、EEPROM上の書き換え頻度を平均化して、EEPROMへの書き換え寿命を延命できるプログラマブル

リードオンリメモリの書き込み回数管理方式を得ることを目的とする。

(問題点を解決するための手段)

この発明に係るプログラマブルリードオンリメモリの書き込み回数管理方式は、記憶領域を複数のブロックに分割し、各ブロック毎に書き込み回数を記憶し、あらかじめ設定される書き込み回数を越えたブロックへの書き込みを抑制させる。

(作用)

この発明においては、記憶領域を各ブロック毎に書き込み回数を記憶しておき、この書き込み回数があらかじめ設定される書き込み回数を越えたら、そのブロックへの書き込みを抑制させる。

(実施例)

第1図(a)はこの発明の一実施例を示すプログラマブルリードオンリメモリへの書き込み回数管理方式を説明する模式図であり、1はEEPROMで、例えば書き込み容量が32768バイト×8ビットで、書き込み回数が1万回に設定してある。EEPROM1は、ポインタブロック1aお

よび予備ポインタブロックSPB1~SPB50より構成される。ポインタブロック1aは4アドレス(各1バイト)で構成され、「0~1」番地の2バイトで、書き換え回数WCNT、例えば「13881₁₆」を記憶している。またポインタブロック1aの「2」番地の1バイトは、ディレクトリDB、例えば「01₁₆」を記憶している。さらに、ポインタブロック1aの「3」番地の1バイトは、未使用のスタートブロック番号OSB、例えば「33₁₆」を記憶している。またポインタブロック1aの「4」番地の1バイトは、未使用のエンドブロック番号OEB、例えば「8A₁₆」を記憶している。

第1図(b)はこの発明の装置構成の一例を説明するブロック図であり、11はCPUで、ROM11a、RAM11bを有し、ROM11aに格納された第6図に示すフローに準じたプログラムに応じて各部を制御する。12は入力手段で、データ書き込み装置13にセットされるEEPROM1へのデータ書き込みおよびデータ消去を指

示する。なお、CPU11にはデータの伝送を行うアキュムレータACC、BCCを有している。

第2図は第1図(a)に示すEEPROM1の構造を示す模式図であり、21はブロック番号であり、例えば127個のブロックBLOCK1~BLOCK127に分割されている。各ブロックは、例えば256バイトで構成され、先頭の2バイトで、そのブロックが更新された回数、すなわち、後述する更新回数が記憶されている。次に続く253バイトは記憶データDATAが記憶されており、最後の1バイトは、記憶データDATAがこのブロックに留まるか、または他のブロックに及ぶかどうかを示す離脱ブロックエリアCBがあり、他のブロックに記憶データDATAが及ぶ場合は、離脱ブロックエリアCBには離脱するブロック番号が記憶され、他のブロックに記憶データDATAが及ばない場合は、離脱ブロックエリアCBには「FF₁₆」が記憶されている。

第3図は第2図に示すディレクトリブロック構造を説明する模式図であり、30は前記ディレ

クトリDBに指示されるディレクトリブロック、31は前記ディレクトリブロック30の更新カウンタで、例えば2バイトで構成される。32はファイル領域で、各ファイル名が12バイトで記憶される。33はスタートブロック番号エリア(SB)で、例えば1バイトで構成され、ファイルのスタートブロック番号が記憶されている。34はエンドブロック番号エリア(EB)で、例えば1バイトで構成され、ファイルのエンドブロック番号が記憶されている。35はチェーンブロックエリア(CB)で、ディレクトリブロック30に属するディレクトリブロックの有無を記憶する。例えばチェーンブロックエリア35が「FF₁₆」となる。なお、ディレクトリブロック30は、例えば18個のファイル領域32で構成される。

次に第1図(a)および第3図を参照しながらEEPROM1の構造について説明する。

第1図(a)に示すようにポインタブロック1aの書き換え回数WCNTに、例えば「1388₁₆」が記憶されているとすると、5000回の

ア34が「18₁₆」となっているため、ブロックBLOCK21から始まり、ブロックBLOCK24で終ることになる。またファイル領域32のファイル3の次に「FF₁₆」が書かれているので、このファイル領域32はファイル3で終了していることになる。

第4図は未使用のEEPROM1の状態を説明する模式図であり、第1図(a)、第3図と同一のものには同じ符号を付している。

この図から分かるように、未使用のEEPROM1のポインタブロック1aの書き換え回数WCNTが「0001₁₆」、ディレクトリDBが「01₁₆」、未使用のスタートブロック番号OSBが「02₁₆」、未使用のエンドブロック番号OEBが「7A₁₆」がそれぞれポインタブロック1aの0番地から4番地にそれぞれ記憶されている。これにより、ディレクトリDBに指示されるブロックBLOCK1を参照すると、更新カウンタ31に「0001₁₆」が書き込まれているとともに、ファイル領域32のファイル1に「FF₁₆」が書

き込まれたことを示し、またディレクトリDBには「01₁₆」が記憶されているので、ディレクトリDBに指示されるディレクトリブロック30のブロック番号が「1」で、そのディレクトリブロック30の更新カウンタ31には、「142FF₁₆」が記憶されている。これは、このディレクトリブロック30を5167回更新したことを示し、ファイル領域32のファイル(File)1(ファイル名)はスタートブロック番号エリア33が「02₁₆」で、エンドブロック番号エリア34が「05₁₆」となっているため、ブロックBLOCK2から始まり、ブロックBLOCK5で終ることになる。またファイル領域32のファイル2は、スタートブロック番号エリア33が「0A₁₆」で、エンドブロック番号エリア34が「0F₁₆」となっているため、ブロックBLOCK10から始まり、ブロックBLOCK15で終ることになる。さらに、ファイル領域32のファイル3(ファイル名)は、スタートブロック番号エリア33が「15₁₆」で、エンドブロック番号エリ

ア35に「FF₁₆」が書き込まれており、EEPROM1が未使用状態であることを示している。

さらに、ポインタブロック1aのスタートブロック番号OSBおよびエンドブロック番号OEBには「02₁₆」、「7F₁₆」がそれぞれ書き込まれている。すなわち、ブロックBLOCK2～127には先頭の2バイトに「0001₁₆」が書き込まれ、最終の1バイトに各後続のブロックの属するチェーンブロックエリア35には、ブロックBLOCK2～128に対して「03～7F₁₆」が書き込まれ、ブロックBLOCK127のチェーンブロックエリア35には「FF」が書き込まれている。このように、各ブロックBLOCK2～127は1つのチェーン構造となる。

次に第3図、第5図(a)、(b)を参照しながらEEPROM1への書き込み動作を説明する。

第5図(a)、(b)はEEPROM1への書き込み動作を説明する模式図であり、第1図

(a)、第3図と同一のものには同じ符号を付している。なお、書き込み直前は、第3図に示す状態であったものとする。

まず、各ブロックBLOCKのファイル領域32の先頭が「0011」のところを探し当てる。第3図の場合は、ファイル2とファイル3との間に「0011」があり、そこにファイル4という名前を12バイトで書き込み、ポインタブロック1aの未使用ブロックのスタートブロック番号OSBの指示するブロックBLOCK、すなわち「5711」の先頭の2バイト情報、すなわち、更新カウンタ31を「1」インクリメントし、その加算値が、例えば1万回を越えているようであれば、ファイル4のチェーンブロックエリア35が示すブロックBLOCKに対して同様の操作を行い、更新カウンタ31が1万回以下のブロックBLOCKを探し当てて、そのブロックBLOCKの番号をポインタブロック1aのスタートブロック番号OSBに書き込むとともに、ファイル4のデータ

をブロックBLOCK87(253バイト)に書き込み、ブロックBLOCK87に達するように、ブロックBLOCK87のチェーンブロックエリア35の指示するブロックBLOCKの更新カウンタ31を「1」インクリメントして加算値が、例えば1万回を越えているかどうかを調べ、指示されるブロックBLOCKの更新カウンタ31が1万回を越えるようであれば、更新回数が1万回以下のブロックBLOCKを探し当て、そのブロックBLOCKの番号を直前に書き込んだブロックBLOCKのチェーンブロックエリア35に書き込む。このようにして、データの書き込みが行われ、更新回数が1万回を越えるブロックBLOCKが排除されて行く。そして、書き込みデータがなくなるまで同様の操作を行い、最後に書き込んだブロックBLOCKのチェーンブロックエリア35に記憶されていた内容を新しい未使用のスタートブロック番号OSBに書き換え、ポインタブロック1aの書き換え回数WCNTを「1」インクリメントして「138911」とな

り、最後にデータを書き込んだブロックBLOCKのチェーンブロックエリア35を「FF11」にする。そして、ディレクトリブロック30の最終ブロック番号を記憶するエンドブロック番号エリア34に最後のデータを書き込んだブロックBLOCKの番号を書き込むとともに、更新カウンタ31を「1」インクリメントすると、第5図(b)に示されるように、更新カウンタ31が「143011」となり、ファイル4のスタートブロック番号エリア33が「3311」で、エンドブロック番号エリア34が「3711」となる。

次に第5図(a)、(b)を参照しながらEEPROM1に書き込まれているファイル1の削除動作について説明する。

ディレクトリブロック30となるブロックBLOCK1よりファイル1を探し、ファイル領域32の先頭の2バイトを「0011」とする。次いで、ディレクトリブロック30の更新カウンタ31を「1」インクリメントし、ファイル1のスタートブロック番号エリア33とエンドブロック

番号エリア34のデータを参照して、ポインタブロック1aのエンドブロックOEBが指示するブロックのチェーンブロックエリア35の内容(削除直前までは「FF11」であった)をスタートブロック番号エリア33の内容に変更し、このブロックの更新カウンタ31を「1」インクリメントする。すなわち、未使用ブロックの最後に今削除したファイル4を接続するわけである。このようにして、更新カウンタ31を進めながら何度もファイルの更新、削除を実行して行くうちに、更新カウンタ31が1万回に接近する。

次に更新カウンタ31が1万回に到達した場合のアクセス処理について説明する。

まず、ポインタブロック1aのスタートブロック番号OSBの内容が示しているブロックBLOCKのチェーンブロックエリア35の内容を新規のスタートブロック番号OSBとする。次いで、このブロック直前のディレクトリブロック30の更新カウンタ31の情報以外の内容を転送する。そして、ポインタブロック1aのディレクトリD

Bに新規のディレクトリブロック番号を書き込み、ポインタブロック1aの書き換え回数WCNTおよび更新カウンタ31を「1」インクリメントする。

一方、ポインタブロック1aの書き換え回数WCNTは1万回を越えた場合は、予備ポインタブロックSPB1~SPB50のうち一番近い予備ポインタブロックへ書き換え回数WCNTの情報以外のデータを転送し、新規のポインタブロックの書き換え回数WCNT(00001a)を「1」インクリメントして「00011a」に設定する。この場合、破棄されたポインタブロック1aの書き換え回数WCNTは1万回以上となり、新のポインタブロック1aの書き換え回数WCNTは1万回以下となる。このようにして、カウンタブロック30およびポインタブロック1aの書き込み削除を管理する。また削除されたファイルが使用していたブロックは未使用ブロックの一番最後に回される。これは、未使用ブロックの使用回数を平均化するためである。

ACCが指示するブロックの容量が235バイトを越えるかどうかを判断し(8)、YESならばアキュムレータACCが指示するブロックの隣接ブロックエリアCBをアキュムレータBCCに記憶させる(9)。次いで、アキュムレータBCCが指示するブロックの書き換え回数WCNTを+1更新する(10)。次いで、書き換え回数WCNTが10000を越えたかどうかを判断し(11)、YESならばアキュムレータBCCの指示するブロックの隣接ブロックエリアCBを記憶させ(12)、ステップ(10)に戻り、NOならばアキュムレータACCが指示するブロックの隣接ブロックエリアCBにアキュムレータBCCの内容を書き込み(13)、ステップ(7)に戻る。

一方、ステップ(8)の判断でNOの場合は、アキュムレータACCが指示する隣接ブロックエリアCBを未使用のスタートブロック番号OSBに書き込む(14)。次いで、ポインタブロック1aの書き換え回数WCNTを+1更新する(15)。次いで、アキュムレータACCが指示するブロックの

第6図は第1図(a)に示したEEPROM1のデータ書き込み制御動作を説明するためのフローチャートである。なお、(1)~(18)は各ステップを示す。

まず、ディレクトリブロック30の空エリアを探して、新規のファイル名を書き込む(1)。次いで、未使用のスタートブロック番号OSBをCPU1のアキュムレータACCに記憶させる(2)。アキュムレータACCが指示するブロックの書き換え回数WCNTを+1更新する(3)。ここで、書き換え回数WCNTが10000を越えたかどうかを判断し(4)、YESならばアキュムレータACCの指示するブロックの隣接ブロックエリアCBをアキュムレータACCに記憶し(5)、ステップ(3)に戻り、NOならばディレクトリブロック30のスタートブロック番号エリア(SB)33にアキュムレータACCの内容を書き込む(6)。次いで、アキュムレータACCが指示するブロックのデータエリアにデータを書き込む(7)。ここで、書き込みデータがアキュムレ-

隣接ブロックエリアCBへ「FFH」を書き込む(16)。そして、ディレクトリブロック30の新ファイル位置のエンドブロック番号エリア34へアキュムレータACCの内容を書き込む(17)。次いで、ディレクトリブロック30の書き換え回数WCNTを更新する(18)。

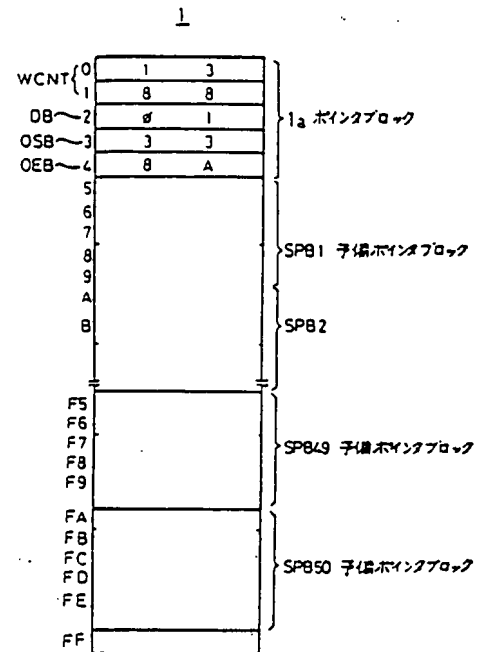
(発明の効果)

以上説明したように、この発明は記憶領域を複数のブロックに分割し、各ブロック毎に書き込み回数を記憶し、あらかじめ設定される書き込み回数を越えたブロックへの書き込みを抑制させるようにしたので、EEPROMに書き込まれるデータの消失を未然に防げるとともに、不要になったブロックを未使用ブロックの最後尾に接続するようにしたので、各ブロックの書き込み回数を平均化できる利点を有する。

4. 図面の簡単な説明

第1図(a)はこの発明の一実施例を示すプログラマブルリードオンリメモリへの書き込み回数管理方式を説明する模式図。第1図(b)はこの

第 1 図 (a)

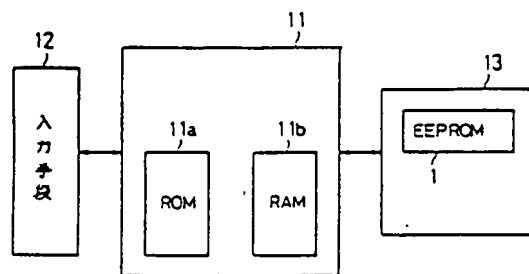


発明の装置構成の一例を説明するブロック図、第2図は第1図(a)に示すEEPROMの構造を示す模式図、第3図は第2図に示す各ディレクトリブロック構造を説明する模式図、第4図は未使用のEEPROM状態を説明する模式図、第5図(a)、(b)はEEPROMへの書き込み動作を説明する模式図、第6図は第1図(a)に示したEEPROMのデータ書き込み動作を説明するためのフローチャートである。

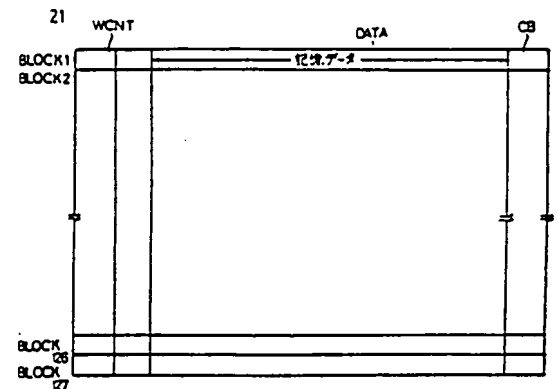
図中、1はEEPROM、1aはポインタブロック、21はブロック番号、30はディレクトリブロック、31は更新カウンタ、32はファイル領域、33はスタートブロック番号エリア、34はエンドブロック番号エリア、35はチェーンブロックエリアである。

代理人 小林 高 司

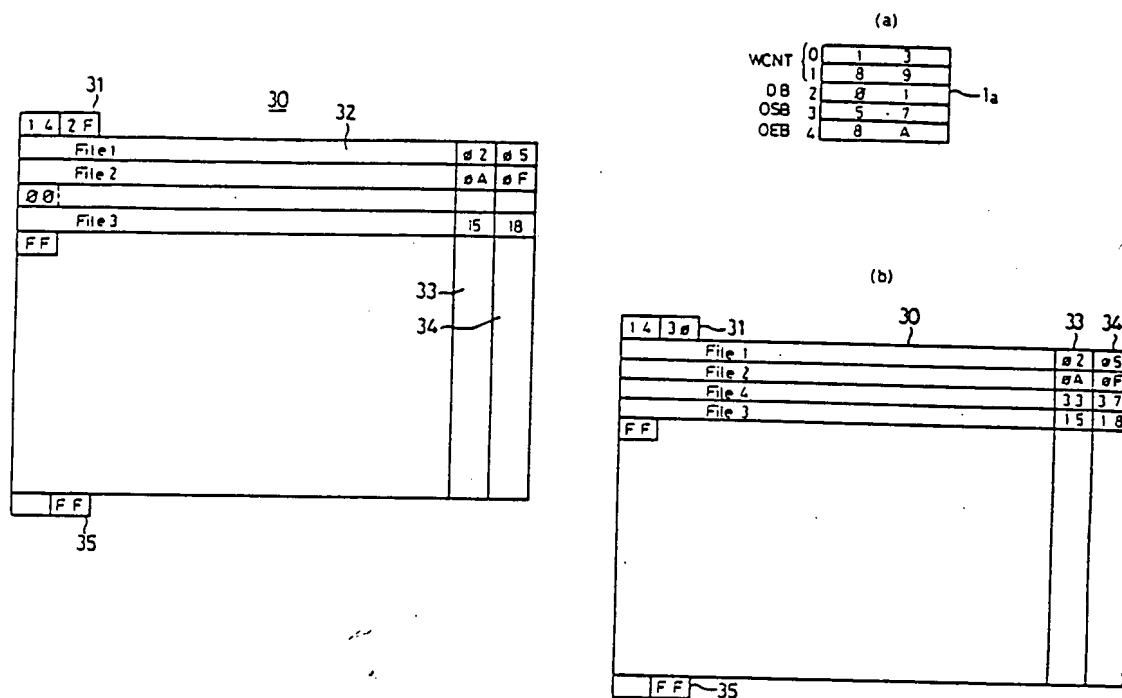
第 1 図 (b)



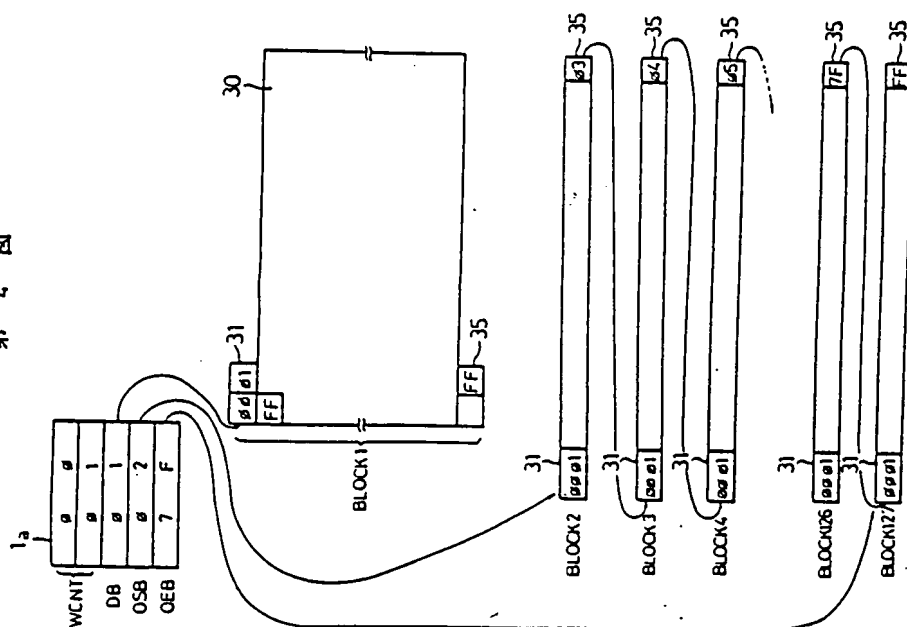
第 2 図



第 5 図



第 4 図



第 6 図

